UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/729,508	12/04/2000	Rene Vangemert	00-487/1496.00053	9954

24319 7590 10/21/2003

LSI LOGIC CORPORATION
1621 BARBER LANE
MS: D-106 LEGAL
MILPITAS, CA 95035

EXAMINER	
GERSTL, SHANE F	

ART UNIT	PAPER NUMBER
2183	5

DATE MAILED: 10/21/2003

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/729,508	VANGEMERT ET AL.	
	Examiner	Art Unit	
	Shane F Gerstl	2183	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *28 February 2001 and 16 August 2001* .

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-15* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-15* is/are rejected.

7)☒ Claim(s) *1, 5, 8, and 12* is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *04 December 2000* is/are: a)☐ accepted or b)☒ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

11)☐ The proposed drawing correction filed on _____ is: a)☐ approved b)☐ disapproved by the Examiner.

If approved, corrected drawings are required in reply to this Office action.

12)☐ The oath or declaration is objected to by the Examiner.

Priority under 35 U.S.C. §§ 119 and 120

13)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____ .

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

14)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application).

a) ☐ The translation of the foreign language provisional application has been received.

15)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449) Paper No(s) _____ .
4) ☐ Interview Summary (PTO-413) Paper No(s). _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: .

DETAILED ACTION

1. Claims 1-15 have been examined.

Papers Received

2. Receipt is acknowledged of drawings and change of address papers submitted, where the papers have been placed of record in the file.

Specification

3. The disclosure is objected to because of the following informalities: The headings of each section should not be underlined or in boldface type as described in 37 CFR 1.77(c).

4. The title of the invention is not descriptive. A new title is required that is clearly indicative of the invention to which the claims are directed.

The following title is suggested: System for Register Recovery Based on Register Status Bits Used for Stalling the Pipeline Until Valid Data is Retrieved.

Appropriate correction is required.

Drawings

5. The drawings are objected to because in figure 5, element 506 appears to be labeled '500' because of the quality of the drawing. A proposed drawing correction or corrected drawings are required in reply to the Office action to avoid abandonment of the application. The objection to the drawings will not be held in abeyance.

Claim Objections

6. Claims 1 and 8 are objected to because of the following informalities: at letter (A) and number (i), it is unclear what the meaning of loading is. The office is taking the

definition of loading to be writing. However, based on the claim language, it is unknown what the purpose of writing invalid data into a register if it is known to be invalid. Clarification is requested.

7. Claims 1 and 8 are objected to because of the following informalities: at letters (F) and (G) there is mention of "valid data" being obtained and loaded into a register. It is unclear whether this "valid data" means data that is valid or data indicating validity. The office is taking the "valid data" to mean data that is valid based on the specification.

8. Claims 5 and 12 are objected to because of the following informalities: at letter (I) and number (viii), it is unclear what difference, if any, there is between writing data and loading data to the register. The office is taking the terms to mean the same thing based on their function.

Appropriate correction is required.

Claim Rejections - 35 USC § 112

9. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

10. Claims 7 and 14 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

11. Claim 7 recites the limitation "said register" in lines 9, 10, 12, and 13 of the claim. There is insufficient antecedent basis for this limitation in the claim. There are two registers of mention before at the time that these limitations are stated. The office is taking said register to mean the first register of mention.

12. Claim 14 recites the limitation "said register" in lines 5, 6, 9, 12, and 13 of the claim. There is insufficient antecedent basis for this limitation in the claim. There are two registers of mention before at the time that these limitations are stated. The office is taking said register to mean the first register of mention.

Claim Rejections - 35 USC § 102

13. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

14. Claims 1, 3-6, and 15 are rejected under 35 U.S.C. 102(b) as being anticipated by Mirapuri (5,590,294).

15. In regard to claim 1, Mirapuri discloses a method of recovering from loading invalid data into a register within a pipelined processor (figure 5 shows pipelining), the method comprising the steps of:

a. setting a register status for said register to an invalid state in response to loading invalid data into said register; In column 9, lines 35-42, Mirapuri discloses a set of control registers that tell whether a pipeline stage contains a valid or invalid instruction. Because the pipelined processor given by Mirapuri has eight stages and takes eight clock cycles to complete one instruction (column 5, lines 16-18), there are therefore at least eight pipeline registers. These registers hold the data for each stage that the control registers give the status of. Column 10, lines 56-58, show that data is loaded as invalid: "Often,

the stall condition is only detected after parts of the pipeline have advanced using incorrect data."

b. stalling said processor in response to an instruction requiring data buffered by said register and said register status being in said invalid state. As shown above from column 10, lines 56-58, a stall condition is detected after invalid data has advanced, showing that the data must be needed since validity was not detected until now. Then after detecting the stall condition, Mirapuri stalls.

16. In regard to claim 3, Mirapuri discloses the method of claim 1, as described above, further comprising the step of: setting said register status for said register to said invalid state in response to receiving data from another register having said invalid state. The case given above for claim 1 shows that if the pipeline has advanced invalid data, then whatever stage where the invalid data was detected has received its data from the previous pipeline stage and register.

17. In regard to claim 4, Mirapuri discloses the method of claim 1, as described above, further comprising the steps of:

a. obtaining valid data for said register from a memory in response to loading invalid data into said register; As described above, invalid data is at times loaded into the registers and when a register is invalid and accessed, the processor stalls. Mirapuri has shown in column 10, lines 63-65 that stages invalid at the interlock (stall) are preloaded with correct information. Because Mirapuri has shown in column 6, lines 54-56, one of multiple cases where correct data is

loaded from memory, it has been shown that Mirapuri loads correct information from memory when invalid data exists.

b. stalling said processor in response to obtaining valid data for said register; In column 10, lines 59-63, Mirapuri has shown that parts of the pipeline are frozen (stalled) while others obtain corrected information. After receiving the corrected data, the pipeline registers of these stages are redone, but, the stalled stages continue to stall, thus in response to receiving valid data.

c. loading said register with valid data in response to stalling said processor in step (F). As just described, when parts of the pipeline are stalled and others receive data, that data is then loaded to the pipeline registers.

18. In regard to claim 5, Mirapuri has disclosed the method of claim 4, as described above further comprising the steps of:

a. stalling said processor prior to writing new data to said register having said invalid state while obtaining said valid data for said register; As applied to claim 4, Mirapuri has disclosed a process that stalls if a register is invalid while obtaining valid data for it. Consequently, this valid data is obtained, and thus the pipeline stalled, before the data can be written to the register.

b. writing new data to said register in response to loading said register with valid data. When valid data is written to the pipeline register of mention in Mirapuri, the pipeline continues its operation as normal and thus as the next instruction passes through each stage, new data is written to the register.

19. In regard to claim 6, Mirapuri discloses the method of claim 1, as described above, further comprising the step of: buffering said register status as a plurality of bits to provide for multiple conditions that would set said register status to said invalid state. In column 9, lines 35-42, Mirapuri discusses the use of the control or status registers. He shows that a status register is used for each stage (pipeline register) to show the validity of the instructions held in the registers using the information stored in each. Because Mirapuri discloses that information rather than a bit is used to determine validity as well as the common practice that a register stores multiple bits, it can be seen that Mirapuri uses multiple bits in each status register to determine validity based on multiple conditions.

20. In regard to claim 15, Mirapuri discloses a pipelined processor (figure 5 shows pipelining) comprising:

a. means for buffering data; The pipelined processor given by Mirapuri has eight stages and takes eight clock cycles to complete one instruction (column 5, lines 16-18), there are therefore at least eight pipeline registers. These registers buffer data to the next stage of the pipeline.

b. means for buffering a register status; In column 9, lines 35-42, Mirapuri discloses a set of control registers that that tell whether a pipeline stage contains a valid or invalid instruction. The pipeline registers mentioned above contain the valid or invalid instruction. Thus, the control registers buffer the status of the pipeline registers.

c. means for setting said register status to an invalid state in response to loading invalid data into said means for buffering data; It is inherent that if the status registers of mention above hold validity data, then there is a means for setting the status. Column 10, lines 56-58, show that data is loaded as invalid when it says, "Often, the stall condition is only detected after parts of the pipeline have advanced using incorrect data."

d. means for stalling said processor in response to an instruction requiring data buffered by said means for buffering data and said register status being in said invalid state. In column 6, lines 43-55, Mirapuri discloses the case of an instruction cache miss. Mirapuri states here that if there is a miss, an interlock occurs resulting in a stall. This miss detection is shown to not occur until after the fetch and decode stages (lines 49-52). Therefore, the pipeline registers of the fetch and decode stage are invalid when a miss occurs as Mirapuri shows, "Such stages are invalid at the time of the interlock," (column 10, lines 63-64).

Claim Rejections - 35 USC § 103

21. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

22. Claims 2 and 7 are rejected under 35 U.S.C. 103(a) as being unpatentable over Mirapuri in view of Sites (5,193,167).

23. In regard to claim 2,

a. Mirupari discloses the method of claim 1, as described above, further comprising the step of: setting said register status to said invalid state (as described above).

b. Mirupari does not disclose the above function in response to a conditional store for said register.

c. Sites has taught in column 3, lines 47-56 the use of a conditional store instruction. This instruction is shown to correspond to a matching locked load instruction. The store updates a memory position from the register loaded by the locked load instruction only if the register was not written to since the locked load instruction. In other words, the store conditional stores the contents of a register into memory based on the register's validity. This means that if the register was written to, the data in the register is not written to memory. Therefore, invalid data was loaded into the register. When one incorporates this disclosure of Sites into the design of Mirapuri, one of ordinary skill in the art can see that when a conditional store is encountered and it is discovered that invalid data was loaded into the register, the status of the register would be set to invalid as shown above.

d. Sites has taught that this conditional store instruction permits the use of atomic byte writes (lines 47-49). This means that outside processes can only access the data in use before and after the transaction is complete. Otherwise, the store is not carried out. This data integrity that is provided during the execution of the conditional store instruction would have motivated one of

ordinary skill in the art to implement the conditional store instruction taught by Sites in the design of Mirapuri.

It would have been obvious to one of ordinary skill in the art at the time of invention to modify the design of Mirapuri to include the conditional store instruction taught by Sites in order to achieve data integrity upon executing a store.

24. In regard to claim 7, Mirapuri has disclosed the method of claim 1, as described above, further comprising the steps of:

a. setting said register status for said register to said invalid state in response to a conditional store for said register.

i. Mirupari discloses the method of claim 1, as described above, further comprising the step of: setting said register status to said invalid state (as described above).

ii. Mirupari does not disclose that said register status for said register is set to said invalid state in response to a conditional store for said register.

iii. Sites has taught in column 3, lines 47-56 the use of a conditional store instruction. This instruction is shown to correspond to a matching locked load instruction. The store updates a memory position from the register loaded by the locked load instruction only if the register was not written to since the locked load instruction. In other words, the store conditional stores the contents of a register into memory based on the register's validity. This means that if the register was written to, the data

in the register is not written to memory. Therefore, invalid data was loaded into the register. When one incorporates this disclosure of Sites into the design of Mirapuri, one of ordinary skill in the art can see that when a conditional store is encountered and it is discovered that invalid data was loaded into the register, the status of the register would be set to invalid as shown above.

iv. Sites has taught that this instruction pair permits the use of atomic byte writes (lines 47-49). This means that outside processes can only access the data in use before and after the transaction is complete. Otherwise, the store is not carried out. This data integrity that is provided during the execution of the conditional store instruction would have motivated one of ordinary skill in the art to implement the conditional store instruction taught by Sites in the design of Mirapuri.

It would have been obvious to one of ordinary skill in the art at the time of invention to modify the design of Mirapuri to include the conditional store instruction taught by Sites in order to achieve data integrity upon executing a store.

b. setting said register status for said register to said invalid state in response to receiving data from another register having said invalid state. The case given above for claim 1 shows that if the pipeline has advanced invalid data, then whatever stage where the invalid data was detected has received its data from the previous pipeline stage and register.

c. obtaining valid data for said register from a memory in response to loading invalid data into said register; As described above, invalid data is at times loaded into the registers and when a register is invalid and accessed, the processor stalls. Mirapuri has shown in column 10, lines 63-65 that stages invalid at the interlock (stall) are preloaded with correct information. Because Mirapuri has shown in column 6, lines 54-56, one of multiple cases where correct data is loaded from memory, it has been shown that Mirapuri loads correct information from memory when invalid data exists.

d. stalling said processor in response to obtaining valid data for said register; In column 10, lines 59-63, Mirapuri has shown that parts of the pipeline are frozen (stalled) while others obtain corrected information. After receiving the corrected data, the pipeline registers of these stages are redone, but, the stalled stages continue to stall, thus in response to receiving valid data.

e. loading said register with valid data in response to stalling said processor in step (F). As just described, when parts of the pipeline are stalled and others receive data, that data is then loaded to the pipeline registers.

25. Claims 8 and 10-13 are rejected under 35 U.S.C. 103(a) as being unpatentable over Mirapuri in view of Steiss (5,815,420).

26. In regard to claim 8,

a. Mirapuri discloses a pipelined processor (figure 5 shows pipelining), comprising:

i. A register configured to buffer data

ii. logic configured to

set said register status for said register to an invalid state in response to loading invalid data into said register; In column 9, lines 35-42, Mirapuri discloses a set of control registers that that tell whether a pipeline stage contains a valid or invalid instruction. Because the pipelined processor given by Mirapuri has eight stages and takes eight clock cycles to complete one instruction (column 5, lines 16-18), there are therefore at least eight pipeline registers. These registers hold the data for each stage that the control registers give the status of. Column 10, lines 56-58, show that data is loaded as invalid: "Often, the stall condition is only detected after parts of the pipeline have advanced using incorrect data."

stall said processor in response to an instruction requiring data buffered by said register and said register status being in said invalid state. As shown above from column 10, lines 56-58, a stall condition is detected after invalid data has advanced, showing that the data must be needed since validity was not detected until now. Then after detecting the stall condition, Mirapuri stalls.

b. Mirapuri does not disclose that the register is configured to buffer a register status. Mirapuri keeps track of register status in control registers that contain pipeline register validity information.

c. Steiss has disclosed a register configured to buffer a register status (figure 2, V bit). This status is used in the same manner as disclosed by Mirapuri for stalling the processor (Steiss, column 10, lines 9-11).

d. Having the register status bits buffered by the registers themselves allows for closer physical access of validity information to its corresponding data. This will then correspond to some speedup because of the physical closeness of needed information. This physical convenience and speedup would have motivated one of ordinary skill in the art at the time of invention to integrate the validity information of the registers in Mirapuri's disclosure into the register itself as described by Steiss.

It would have been obvious to one of ordinary skill in the art at the time of invention to modify the design of Mirapuri to include the register status information in the register itself as given by Steiss for the purposes of convenience in location and logical speedup.

27. In regard to claim 10, Mirapuri in view of Steiss discloses the pipelined processor of claim 8, as described above, further comprising: another register; said logic being further configured to set said register status for said register to said invalid state in response to receiving data from another register having said invalid state. The case given above for claim 8 shows that if the pipeline has advanced invalid data, then whatever stage where the invalid data was detected has received its data from the previous pipeline stage and register.

28. In regard to claim 11, Mirapuri in view of Steiss discloses the pipelined processor of claim 8, as described above, further comprising:

a. A bus interface unit (figure 4, element 14) configured to obtain valid data for said register from a memory in response to loading invalid data into said register; The memory management unit controls accesses to and from the memory thus controlling the bus and acting as a bus interface unit. As described above, invalid data is at times loaded into the registers and when a register is invalid and accessed, the processor stalls. Mirapuri has shown in column 10, lines 63-65 that stages invalid at the interlock (stall) are preloaded with correct information. Because Mirapuri has shown in column 6, lines 54-56, one of multiple cases where correct data is loaded from memory, it has been shown that Mirapuri loads correct information from memory when invalid data exists.

b. Said logic being further configured to

i. stall said processor in response to obtaining valid data for said register; In column 10, lines 59-63, Mirapuri has shown that parts of the pipeline are frozen (stalled) while others obtain corrected information. After receiving the corrected data, the pipeline registers of these stages are redone, but, the stalled stages continue to stall, thus in response to receiving valid data.

ii. load said register with valid data in response to stalling said processor; As just described, when parts of the pipeline are stalled and others receive data, that data is then loaded to the pipeline registers.

29. In regard to claim 12, Mirapuri in view of Steiss has disclosed the pipelined processor of claim 11, as described above, further comprising: said logic being further configured to

a. stall said processor prior to writing new data to said register having said invalid state while obtaining said valid data for said register; As applied to claim 4, Mirapuri has disclosed a process that stalls if a register is invalid while obtaining valid data for it. Consequently, this valid data is obtained, and thus the pipeline stalled, before the data can be written to the register.

b. writing new data to said register in response to loading said register with valid data obtained from said. When valid data from memory is written to the pipeline register of mention in Mirapuri indicating that the stall is completed, the pipeline continues its operation as normal and thus as the next instruction passes through each stage, new data is written to the register.

30. In regard to claim 13,

a. Mirapuri in view of Steiss, as applied to claim 8, discloses the pipelined processor of claim 8, as described above, that includes the register status in control registers that are checked on each clock cycle for indicating invalid states of pipeline stages;

b. Mirapuri in view of Steiss, as applied to claim 8, does not disclose that the register status is buffered as a plurality of bits to provide for multiple conditions that would set said register status to said invalid state.

c. Steiss has disclosed a pipelined processor (figure 1) that keeps track of register status. Steiss has disclosed that register status is buffered as a plurality of bits (figure 3, element 51) to provide for multiple conditions that would set said register status to said invalid state. In column 6, lines 50-56, Steiss shows the operation of the mentioned buffer as holding the status of multiple operands (conditions). In column 10, lines 9-11, Steiss has shown that these valid bits are used for stalling the processor, as is so in the disclosure of Mirapuri in view of Steiss as applied to claim 8.

d. Implementing the multiple bit validity buffer of Steiss in the design of Mirapuri in view of Steiss, as applied to claim 8, would allow each pipeline stage to realize invalid information for multiple conditions based on different operands. Upon realizing that the stage has received invalid data, the pipeline register would signify invalid data as described above. The ability to detect multiple reasons for invalidity would have motivated one of ordinary skill in the art to integrate the design of Steiss' validity buffer into that of Mirapuri in view of Steiss, as applied to claim 8.

It would have been obvious to one of ordinary skill in the art at the time of invention to modify the design of Mirapuri in view of Steiss, as applied to claim 8, to include the validity buffer of Steiss in order to obtain the ability to detect multiple reasons for pipeline stage invalidity.

31. Claims 9 and 14 are rejected under 35 U.S.C. 103(a) as being unpatentable over Mirapuri in view of Steiss as applied to claim 8 above, and further in view of Sites.

32. In regard to claim 9,

a. Mirupari in view of Steiss discloses the pipelined processor of claim 8, as described above, further comprising: said logic being further configured to set said register status to said invalid state (as described above).

b. Mirupari in view of Steiss does not disclose the above function in response to a conditional store for said register.

c. Sites has taught in column 3, lines 47-56 the use of a conditional store instruction. This instruction is shown to correspond to a matching locked load instruction. The store updates a memory position from the register loaded by the locked load instruction only if the register was not written to since the locked load instruction. In other words, the store conditional stores the contents of a register into memory based on the register's validity. This means that if the register was written to, the data in the register is not written to memory. Therefore, invalid data was loaded into the register. When one incorporates this disclosure of Sites into the design of Mirapuri, one of ordinary skill in the art can see that when a conditional store is encountered and it is discovered that invalid data was loaded into the register, the status of the register would be set to invalid as shown above.

d. Sites has taught that this instruction pair permits the use of atomic byte writes (lines 47-49). This means that outside processes can only access the data in use before and after the transaction is complete. Otherwise, the store is not carried out. This data integrity that is provided during the execution of the

conditional store instruction would have motivated one of ordinary skill in the art to implement the conditional store instruction pair taught by Sites in the design of Mirapuri in view of Steiss.

It would have been obvious to one of ordinary skill in the art at the time of invention to modify the design of Mirapuri in view of Steiss to include the conditional store instruction pair taught by Sites in order to achieve data integrity upon executing a store.

33. In regard to claim 14, Mirapuri in view of Steiss has disclosed the pipelined processor of claim 8, as described above, further comprising:

a. A bus interface unit (figure 4, element 14) configured to obtain valid data for said register from a memory in response to loading invalid data into said register; The memory management unit controls accesses to and from the memory thus controlling the bus and acting as a bus interface unit. As described above, invalid data is at times loaded into the registers and when a register is invalid and accessed, the processor stalls. Mirapuri has shown in column 10, lines 63-65 that stages invalid at the interlock (stall) are preloaded with correct information. Because Mirapuri has shown in column 6, lines 54-56, one of multiple cases where correct data is loaded from memory, it has been shown that Mirapuri in view of Steiss loads correct information from memory when invalid data exists.

b. said logic being further configured to set said register status for said register to said invalid state in response to a conditional store for said register.

i. Mirupari discloses the method of claim 1, as described above, further comprising the step of: setting said register status to said invalid state (as described above).

ii. Mirupari does not disclose the above function in response to a conditional store for said register.

iii. Sites has taught in column 3, lines 47-56 the use of a conditional store instruction. This instruction is shown to correspond to a matching locked load instruction. The store updates a memory position from the register loaded by the locked load instruction only if the register was not written to since the locked load instruction. In other words, the store conditional stores the contents of a register into memory based on the register's validity. This means that if the register was written to, the data in the register is not written to memory. Therefore, invalid data was loaded into the register. When one incorporates this disclosure of Sites into the design of Mirapuri, one of ordinary skill in the art can see that when a conditional store is encountered and it is discovered that invalid data was loaded into the register, the status of the register would be set to invalid as shown above.

iv. Sites has taught that this instruction pair permits the use of atomic byte writes (lines 47-49). This means that outside processes can only access the data in use before and after the transaction is complete. Otherwise, the store is not carried out. This data integrity that is provided

during the execution of the conditional store instruction would have motivated one of ordinary skill in the art to implement the conditional store instruction pair taught by Sites in the design of Mirapuri.

It would have been obvious to one of ordinary skill in the art at the time of invention to modify the design of Mirapuri to include the conditional store instruction pair taught by Sites in order to achieve data integrity upon executing a store.

c. another register; set said register status for said register to said invalid state in response to receiving data from said another register having said invalid state. The case given above for claim 1 shows that if the pipeline has advanced invalid data, then whatever stage where the invalid data was detected has received its data from the previous pipeline stage and register.

d. stall said processor in response to obtaining valid data for said register; In column 10, lines 59-63, Mirapuri has shown that parts of the pipeline are frozen (stalled) while others obtain corrected information. After receiving the corrected data, the pipeline registers of these stages are redone, but, the stalled stages continue to stall, thus in response to receiving valid data.

e. load said register with valid data in response to stalling said processor in step (F). As just described, when parts of the pipeline are stalled and others receive data, that data is then loaded to the pipeline registers.

f. set said register to said invalid state in response to a cache load-miss for said register. In column 6, lines 43-54, Mirapuri discloses the case of a cache

miss. Mirapuri shows here that in such a case, the processor stalls. As shown above, the pipeline register is set to invalid when correct data is received by the bus interface unit.

Conclusion

34. The following is text cited from 37 CFR 1.111(c): In amending in reply to a rejection of claims in an application or patent under reexamination, the applicant or patent owner must clearly point out the patentable novelty which he or she thinks the claims present in view of the state of the art disclosed by the references cited or the objections made. The applicant or patent owner must also show how the amendments avoid such references or objections.

35. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure. The following patents are cited to further show the art with respect to stalling a processor based on validity.

US Pat No 5,724,533 to Kuslak shows a method and apparatus for efficiently halting a processor when a cache miss is detected.

US Pat No 6,304,955 to Arora shows the use of register status signals for hazard detection and stalling of the processor.

US Pat No 6,308,261 to Morris shows a computer system with availability status flags for each register that stall the unit when the register is accessed and not available.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Shane F Gerstl whose telephone number is (703)305-7035. The examiner can normally be reached on M-F 8:00-4:30.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (703)305-9712. The fax phone number for the organization where this application or proceeding is assigned is (703) 872-9306.

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is (703)305-3900.

Shane F Gerstl
Examiner
Art Unit 2183

SFG
October 8, 2003

EDDIE CHAN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100